Kauno technologijos universitetas

Informatikos fakultetas

Schemų aprašymas VHDL kalba (skaitikliai)

P175B100 Skaitmeninės logikos pradmenų ketvirtas laboratorinis darbas

Projekto autorius
Gustas Klevinskas

Akademinė grupė
IFF-8/7

Vadovai
Doc. Tomas Adomkus

Kaunas, 2019

Turinys

Įvadas

Darbo tikslas – naudojantis VHDL kalba suprojektuoti skaitiklį, kuris naudodamas tris mažesnius skaitiklius, galėtų skaičiuoti iki duotosios vertės. Patikrinti skaitiklio veikimą programuojamos logikos schemoje.

Užduotys:

1. Pagal duotus reikalavimus aprašyti M1, M2 ir M3 skaitiklius VHDL kalba;
2. Sukurti papildomą failą, kuris apjungtų M1, M2 ir M3 skaitiklius;
3. Parašyti stimulą ir patikrinti suprojektuoto skaitiklio funkcionavimą;
4. Pakoreguoti schemą ir ją įkelti į FPGA plokštę.

Skaitikliai

M1

Šis skaitiklis turi skaičiuoti moduliu 8. Kadangi maksimali skaitiklio vertė bus 7, jam suprojektuoti užteks 3 bitų.

Skaitiklio M1 VHDL failas:

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity M1 is port (
	CLK : in std_logic;		-- Sinchro signalas
	RST : in std_logic;		-- Reset signalas
	CNT_CMD : in std_logic;		-- Komanda
	CNT_C : out std_logic;		-- Pernasa
	CNT_O : out std_logic_vector (2 downto 0));
end M1;

architecture rtl of M1 is
	signal CNT_A : unsigned (2 downto 0);
begin
	process (CLK, RST, CNT_CMD)
	begin
		if RST = '1' then
			CNT_A <= "000";
			CNT_C <= '1';
		elsif CLK'event and CLK = '1' and CNT_CMD = '1' then
			if CNT_A < 7 then	-- Pernasos signalas
				CNT_A <= CNT_A + 1;
				if CNT_A = 6 then
					CNT_C <= '0';
				else
					CNT_C <= '1';
				end if;
			else
				CNT_C <= '1';
				CNT_A <= "000";
			end if;
		end if;
	end process;
CNT_O <= std_logic_vector (CNT_A);
end rtl;
```

M2

Šis skaitiklis turi skaičiuoti moduliu 38. Jam reikės 6 bitų skaitiklio, kadangi $2^5 = 32$ ir tai yra per mažai norint pavaizduoti skaičių 38. O $2^6 = 64$, taigi reikės 6 bitų.

Skaitiklio M2 VHDL failas:

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity M2 is port (
	CLK : in std_logic;		-- Sinchro signalas
	RST : in std_logic;		-- Reset signalas
	CNT_CMD : in std_logic;	-- Komanda
	CNT_C : out std_logic;	-- Pernasa
	CNT_O : out std_logic_vector (5 downto 0));
end M2;

architecture rtl of M2 is
	signal CNT_A : unsigned (5 downto 0);
begin
	process (CLK, RST, CNT_CMD)
	begin
		if RST = '1' then
			CNT_A <= "000000";
			CNT_C <= '1';
		elsif CLK'event and CLK = '1' and CNT_CMD = '1' then
			if CNT_A < 37 then	-- Pernasos signalas
				CNT_A <= CNT_A + 1;
				if CNT_A = 36 then
					CNT_C <= '0';
				else
					CNT_C <= '1';
				end if;
			else
				CNT_C <= '1';
				CNT_A <= "000000";
			end if;
		end if;
	end process;
CNT_O <= std_logic_vector (CNT_A);
end rtl;
```

M3

Šis skaitiklis turi skaičiuoti moduliu 60. Jau anksčiau parodžiau, kad maksimali 6 bitų skaičiaus vertė yra 64, o 5 bitų – 32. Taigi ir M3 skaitikliui reikės 6 bitų.

Skaitiklio M3 VHDL failas:

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity M3 is port (
	CLK : in std_logic;		-- Sinchro signalas
	RST : in std_logic;		-- Reset signalas
	CNT_CMD : in std_logic;	-- Komanda
	CNT_C : out std_logic;	-- Pernasa
	CNT_O : out std_logic_vector (5 downto 0));
end M3;

architecture rtl of M3 is
	signal CNT_A : unsigned (5 downto 0);
begin
	process (CLK, RST, CNT_CMD)
	begin
		if RST = '1' then
			CNT_A <= "000000";
			CNT_C <= '1';
		elsif CLK'event and CLK = '1' and CNT_CMD = '1' then
			if CNT_A < 59 then	-- Pernasos signalas
				CNT_A <= CNT_A + 1;
				if CNT_A = 58 then
					CNT_C <= '0';
				else
					CNT_C <= '1';
				end if;
			else
				CNT_C <= '1';
				CNT_A <= "000000";
			end if;
		end if;
	end process;
CNT_O <= std_logic_vector (CNT_A);
end rtl;
```

JM2

Tai yra skaitiklis, kuriame naudojame visus tris sukurtus skaitiklius. Pagal užduotį, jis turėtų skaičiuoti iki 4560. Apskaičiuokime, ties kuriomis vertėmis turi būti M1, M2 ir M3 skaitikliai, kad JM2 skaičiuotų iki 4560 (*div* – dalyba paimant sveikąją dalį).

$$M1_{fin} = JM2 \; mod \; M1 = 4560 \; mod \; 8 = 0$$

$$M2_{fin} = JM2 \; div \; M1 \; mod \; M2 = 4560 \; div \; 8 \; mod \; 38 = 30$$

$$M3_{fin} = JM2 \; div \; M1 \; div \; M2 = 4560 \; div \; 8 \; div \; 38 = 15$$

Taigi, galutinė M1 reikšmė – 0 (000_2), M2 – 30 (011110_2), M3 – 15 (001111_2).

JM2 aprašymas VHDL kalba:

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity TOP_CNT is port (
       CLK_TOP : in std_logic;                    -- Sinchro signalas
       RST_TOP : in std_logic;                    -- Reset signalas
       Enable_TOP : in std_logic;          -- Aktyvavimo signalas
       Pernasa_TOP : out std_logic);-- Pernasa
end TOP_CNT;

architecture struct of TOP_CNT is
signal C, RST_internal, C1, C2, C3 : std_logic;
signal CNT_1_O : std_logic_vector (2 downto 0);
signal CNT_2_O : std_logic_vector (5 downto 0);
signal CNT_3_O : std_logic_vector (5 downto 0);

component M1 port (
              CLK : in std_logic;          -- Sinchro signalas
              RST : in std_logic;          -- Reset signalas
              CNT_CMD : in std_logic;      -- Aktyvavimo signalas
              CNT_C : out std_logic;       -- Pernasa
              CNT_O : out std_logic_vector (2 downto 0));
end component;

component M2 port (
              CLK : in std_logic;          -- Sinchro signalas
              RST : in std_logic;          -- Reset signalas
              CNT_CMD : in std_logic;      -- Aktyvavimo signalas
              CNT_C : out std_logic;       -- Pernasa
              CNT_O : out std_logic_vector (5 downto 0));
end component;

component M3 port (
              CLK : in std_logic;          -- Sinchro signalas
              RST : in std_logic;          -- Reset signalas
              CNT_CMD : in std_logic;      -- Aktyvavimo signalas
              CNT_C : out std_logic;       -- Pernasa
              CNT_O : out std_logic_vector (5 downto 0));
end component;

begin
       CNT_1 : M1 port map (CLK => CLK_TOP,
              RST => RST_internal, CNT_CMD => Enable_TOP,
              CNT_C => C1, CNT_O => CNT_1_O);
       CNT_2 : M2 port map (CLK => C1,
```

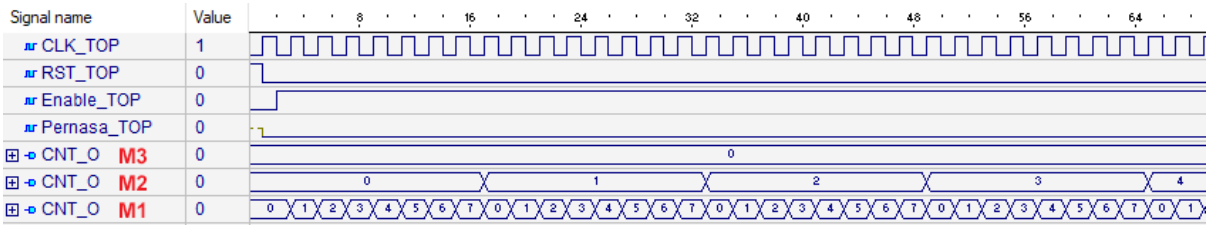
```
		RST => RST_internal, CNT_CMD => Enable_TOP,
		CNT_C => C2, CNT_O => CNT_2_O);
	CNT_3 : M3 port map (CLK => C2,
		RST => RST_internal, CNT_CMD => Enable_TOP,
		CNT_C => C3, CNT_O => CNT_3_O);

	process (CLK_TOP , RST_TOP)
	begin
		if (RST_TOP = '1') then
			RST_internal <= '1';
		elsif CLK_TOP'event and CLK_TOP = '1' then

			-- Stabdymas pasiekus JM2 verte
			if ((CNT_1_O = "000")
			and (CNT_2_O = "011110")
			and (CNT_3_O = "001111")) then
				RST_internal <= '1';
				Pernasa_TOP <= '1';
			else
				RST_internal <= '0';
				Pernasa_TOP <= '0';
			end if;
		end if;
	end process;
end struct;
```

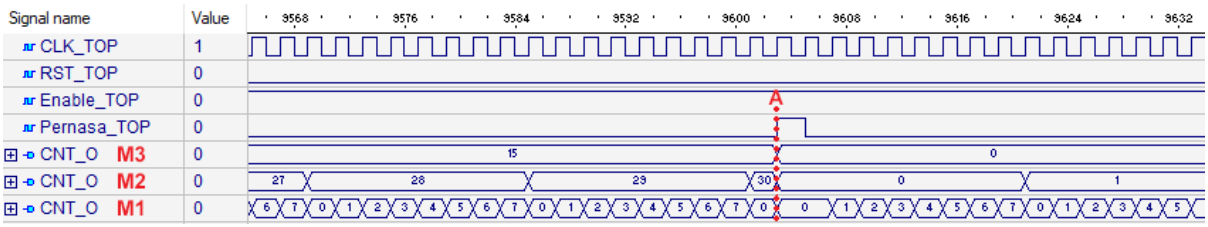
Rezultatai

Pav. 1 matome laiko diagramos pradžią. Kai RST ir Enable signalai yra atitinkamai 0 ir 1, tuomet pradeda veikti skaitiklis. M1 reaguoja į kiekvieną kylantį CLK frontą; M2 padidėja vienetu, kai M1 pasiekia 7; ir, nors čia nesimato, M3 padidėja vienetu, kai M2 pasiekia 38.



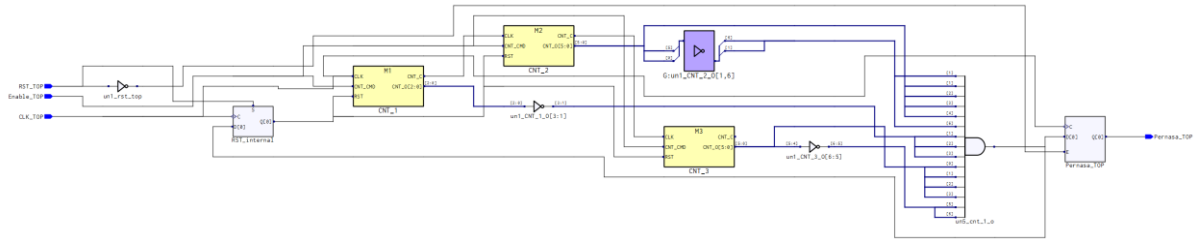
Pav. 1. Laiko diagramos pradžia.

Laiko diagramoje (Pav. 2) raudonai pažymėjau tašką A, ties kuriu M1 yra 0, M2 yra 30 ir M3 yra 15 (skaitiklis pasiekia 4560) ir sugeneruoja pernašos signalą kartu grįždamas į pradinę būseną.



Pav. 2. Laiko diagrama, kai skaitiklis pasiekia JM2.

Paleidus *„Simplify Pro“* įrankį galime pamatyti, kaip atrodo iš VHDL kodo sugeneruota schema.



Pav. 3. Schema, gauta iš „Simplify Pro“ įrankio.

Perėjimas prie FPGA

Į FPGA plokštę kelsiu M1 skaitiklį, nes jo modulis mažiausias ir jį bus lengviausia pavaizduoti.

Tam, kad M1 skaitiklį galėtume įkelti į FPGA plokštę, jo kodą reikės truputį pakoreguoti. Pirmiausia pakeičiau RST signalo sąlygą. Po to reikėjo invertuoti skaitiklio išvesties vektorių, nes FPGA plokštėje LED užsidega, kai signalas būna 0. Taip atrodo pakoreguotas kodas:

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity M1 is port (
	CLK : in std_logic;		-- Sinchro signalas
	RST : in std_logic;		-- Reset signalas
	CNT_CMD : in std_logic;		-- Komanda
	CNT_C : out std_logic;		-- Pernasa
	CNT_O : out std_logic_vector (2 downto 0));
end M1;

architecture rtl of M1 is
	signal CNT_A : unsigned (2 downto 0);
begin
	process (CLK, RST, CNT_CMD)
	begin
		if RST = '0' then
			CNT_A <= "000";
			CNT_C <= '1';
		elsif CLK'event and CLK = '1' and CNT_CMD = '1' then
			if CNT_A < 7 then	-- Pernasos signalas
				CNT_A <= CNT_A + 1;
				if CNT_A = 6 then
					CNT_C <= '0';
				else
					CNT_C <= '1';
				end if;
			else
				CNT_C <= '1';
				CNT_A <= "000";
			end if;
		end if;
	end process;
CNT_O <= not(std_logic_vector (CNT_A));
end rtl;
```

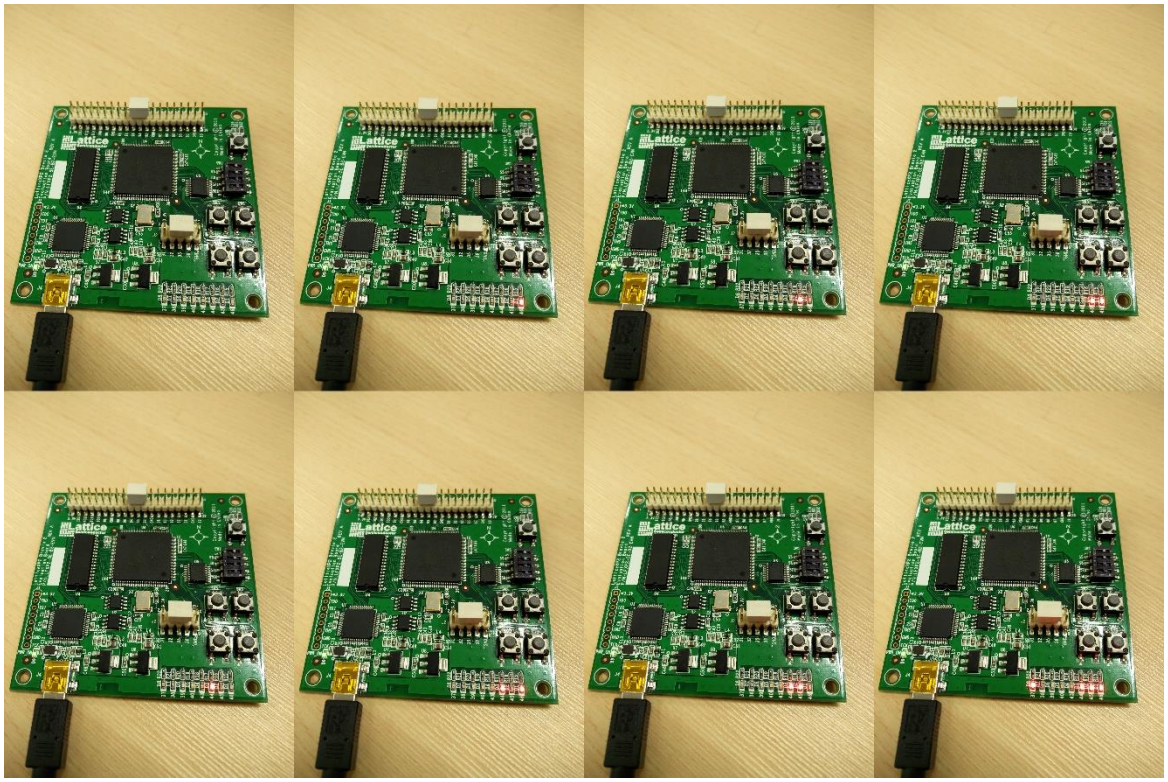
Po to priskiriame skaitiklio įvestis ir išvestis konkretiems fiziniams kontaktams per „Spreadsheet View“ įrankį.

	Name	Group By	Pin	BANK	VREF	IO_TYPE	PULLMODE
1	All Ports	N/A	N/A	N/A	N/A		
1.1	Input	N/A	N/A	N/A	N/A	N/A	N/A
1.1.1	CNT_CMD	N/A	56(56)	5(5)	N/A	LVCMOS2...	UP(UP)
1.1.2	Clock	N/A	N/A	N/A	N/A	N/A	N/A
1.1.2.1	CLK	N/A	53(53)	5(5)	N/A	LVCMOS2...	UP(UP)
1.1.3	RST	N/A	55(55)	5(5)	N/A	LVCMOS2...	UP(UP)
1.2	Output	N/A	N/A	N/A	N/A	N/A	N/A
1.2.1	CNT_C	N/A	37(37)	5(5)	N/A	LVCMOS2...	UP(UP)
1.2.2	CNT_O[0]	N/A	46(46)	5(5)	N/A	LVCMOS2...	UP(UP)
1.2.3	CNT_O[1]	N/A	45(45)	5(5)	N/A	LVCMOS2...	UP(UP)
1.2.4	CNT_O[2]	N/A	44(44)	5(5)	N/A	LVCMOS2...	UP(UP)

Pav. 4. „Spreadsheet View“ langas.

Belieka sugeneruoti kodą ir užprogramuoti FPGA plokštę. Pateiksiu nuotraukas, kuriose matosi plokštės veikimas.

Jungiklyje uždėjau RST ir Enable signalus, kad nereikėtų laikyti nuspaustų mygtukų. Matome, kad kiekvieną kartą paspaudus CLK mygtuką, LED vaizduojamas dvejetainis kodas padidėja vienetu. Paskutinėje nuotraukoje pasiekiama vertė 7 ir sugeneruojamas pernašos signalas (pats kairiausias LED).



Pav. 5. Skaitiklio M1 veikimas FPGA plokštėje.

Išvados

Darbas buvo atliktas sėkmingai; testuojant skaitiklį jis skaičiavo iki reikiamos reikšmės.

Tokią skaitiklio schemą nubraižyti ranka būtų ganėtinai sunku, tačiau pasinaudojus VHDL kalba, šią užduotį galima labai lengvai ir greitai atlikti. Tereikia pakoreguoti duotą kodą. Šis procesas ženkliai pagreitina schemų projektavimo laiką.